16869B-017410US
Atty: George B. F. Yee, Reg. No. 37,478
Telephone: 650-326-2400
Cell-Level Data Access Control Using User-Defined Functions
Inventors: Shinji Fujiwara et al.
Sheets of drawing: 1 of 8

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
2222	1	3	600	500
3333	2	6	900	950
4444	1	2	1200	1500

PT_ID	VST	P_NM	AGE	SEX	MD_ID	DRG	STAY	COST	PYMT
12345	1	AREN	54	M	3333	123	7	1000	1200
23456	1	ERIS	25	F	4444	123	2	1200	1500
12345	2	AREN	55	M	2222	127	3	600	500
97531	1	MARY	65	F	3333	234	5	800	700

FIG. 1

MD_ID
2222

PT_ID	VST	P_NM	AGE	SEX	MD_ID	DRG	STAY	COST	PYMT
▨	▨	▨	54	M	▨	123	7	1000	1200
▨	▨	▨	25	F	▨	123	2	1200	1500
12345	2	AREN	55	M	2222	127	3	600	500
▨	▨	▨	65	F	▨	234	5	800	700

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
2222	1	3	600	900
▨	2	6	900	950
▨	1	2	1200	1500

MD_ID
3333

PT_ID	VST	P_NM	AGE	SEX	MD_ID	DRG	STAY	COST	PYMT
12345	1	AREN	54	M	3333	123	7	1000	1200
▨	▨	▨	25	F	▨	123	2	1200	1500
▨	▨	▨	55	M	▨	127	3	600	500
97531	1	MARY	65	F	3333	234	5	800	700

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
▨	1	3	600	500
3333	2	6	900	950
▨	1	2	1200	1500

MD_ID
4444

PT_ID	VST	P_NM	AGE	SEX	MD_ID	DRG	STAY	COST	PYMT
▨	▨	▨	54	M	▨	123	7	1000	1200
23456	1	ERIS	25	F	4444	123	2	1200	1500
▨	▨	▨	55	M	▨	127	3	600	500
▨	▨	▨	65	F	▨	234	5	800	700

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
▨	1	3	600	500
▨	2	6	900	950
4444	1	2	1200	1500

FIG. 2

```
CREATE VIEW INPT_FACT AS
SELECT (CASE WHEN MD_ID = user-id THEN PT_ID   ELSE NULL END) PT_ID,
    (CASE WHEN MD_ID = user-id THEN VST      ELSE NULL END) VST,
    (CASE WHEN MD_ID = user-id THEN P_NM     ELSE NULL END) P_NM,
    AGE, SEX,
    (CASE WHEN MD_ID = user-id THEN MD_ID  ELSE NULL END) MD_ID,
    DRG, STAY, COST, PYMT
FROM INPT_BASE;
```

FIG. 3

```
SELECT MD_ID, COUNT(*) VOL,
    AVG(STAY) AVG_STAY, AVG(COST) AVG_COST, AVG(PYMT) AVG_PYMT
FROM INPT_FACT
GROUP BY MD_ID ORDER BY AVG_STAY DESC;
```

FIG. 4

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
▨	3	4.67	1000	1133
2222	1	3	600	500

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
3333	2	6	900	950
▨	2	2.5	900	1000

MD_ID	VOL	AVG_STAY	AVG_COST	AVG_PYMT
▨	3	5	800	800
4444	1	2	1200	1500

FIG. 5

```
CREATE VIEW INPT_GRP_BY_MD
SELECT (CASE WHEN MD_ID = user-id THEN MD_ID ELSE NULL END) MD_ID,
      COUNT(*) VOL,
      AVG(STAY) AVG_STAY, AVG(COST) AVG_COST, AVG(PYMT) AVG_PYMT
   FROM INPT_BASE GROUP BY MD_ID ORDER BY AVG_STAY DESC;
```

FIG. 6

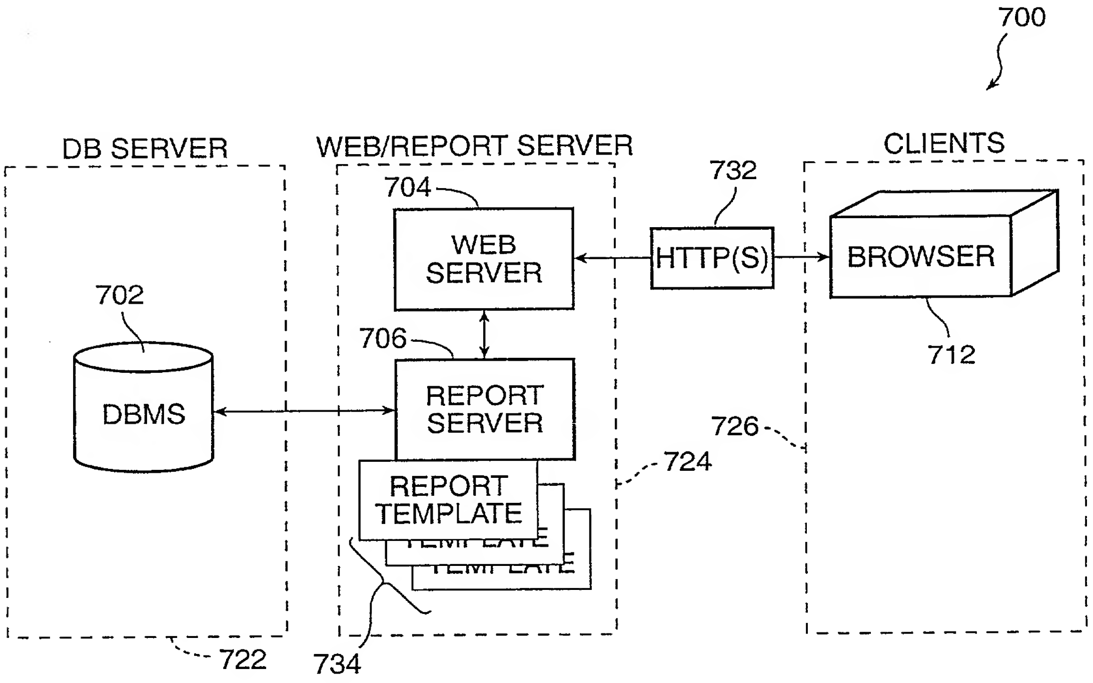
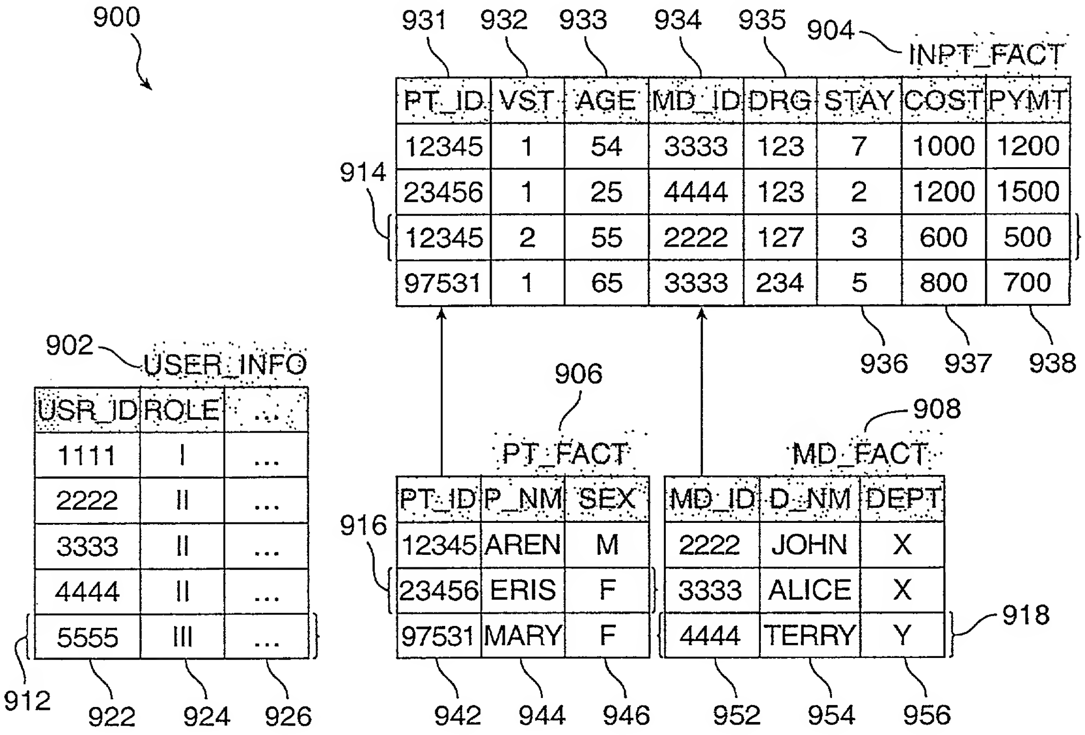


FIG. 7

Access Level	Role	Access Policy
I	Executive	Access to all data.
II	Medical Doctor	Access to doctor's own patient visit data only. The patient privacy information of the other patient visit data should be blinded. The other medical doctor's privacy information should be also blinded.
III	Financial Analyst	Access to financial information without any medical doctor's information.

FIG. 8



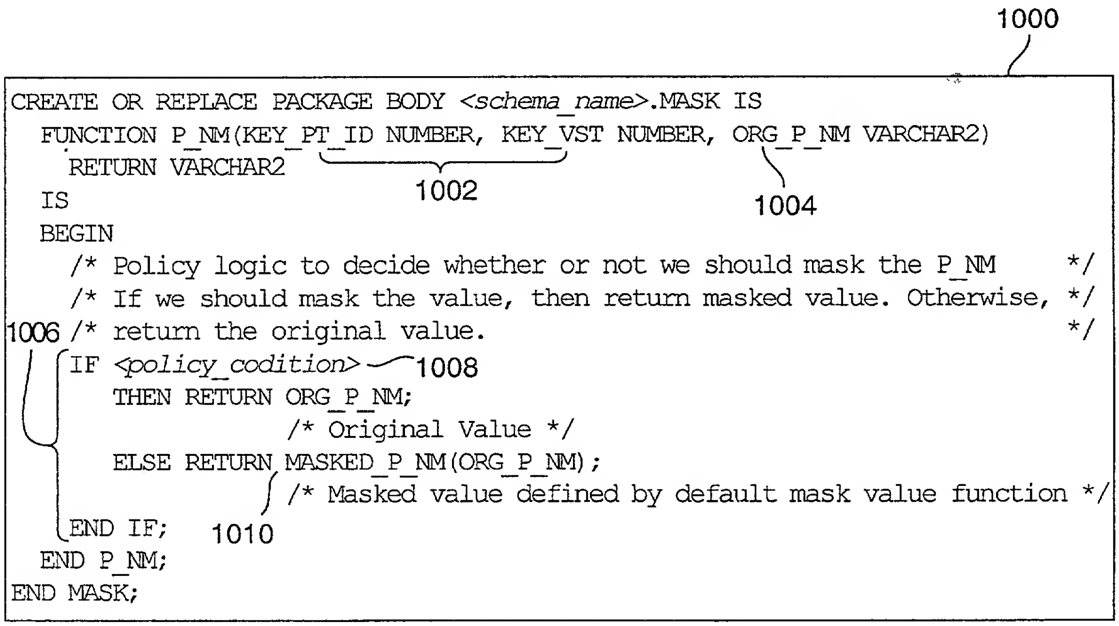
PT_ID	VST	AGE	MD_ID	DRG	STAY	COST	PYMT
12345	1	54	3333	123	7	1000	1200
23456	1	25	4444	123	2	1200	1500
12345	2	55	2222	127	3	600	500
97531	1	65	3333	234	5	800	700

USR_ID	ROLE	...
1111	I	...
2222	II	...
3333	II	...
4444	II	...
5555	III	...

PT_ID	P_NM	SEX
12345	AREN	M
23456	ERIS	F
97531	MARY	F

MD_ID	D_NM	DEPT
2222	JOHN	X
3333	ALICE	X
4444	TERRY	Y

FIG. 9



```
CREATE OR REPLACE PACKAGE BODY <schema_name>.MASK IS
  FUNCTION P_NM(KEY_PT_ID NUMBER, KEY_VST NUMBER, ORG_P_NM VARCHAR2)
    RETURN VARCHAR2
  IS
  BEGIN
    /* Policy logic to decide whether or not we should mask the P_NM      */
    /* If we should mask the value, then return masked value. Otherwise, */
    /* return the original value.                                         */
    IF <policy_codition>
       THEN RETURN ORG_P_NM;
                 /* Original Value */
       ELSE RETURN MASKED_P_NM(ORG_P_NM);
                 /* Masked value defined by default mask value function */
    END IF;
  END P_NM;
END MASK;
```

FIG. 10

```
SELECT   c.PT_ID,
         i.VST,
         p.P_NM,
         p.AGE, p.SEX,
         i.MD_ID,
         m.D_NM,
         i.DRG, i.STAY, ...
  FROM INPT_FACT i, MD_FACT m, PT_FACT p
WHERE  i.PT_ID=p.PT_ID AND i.MD_ID = m.MD_ID AND ... ;
```

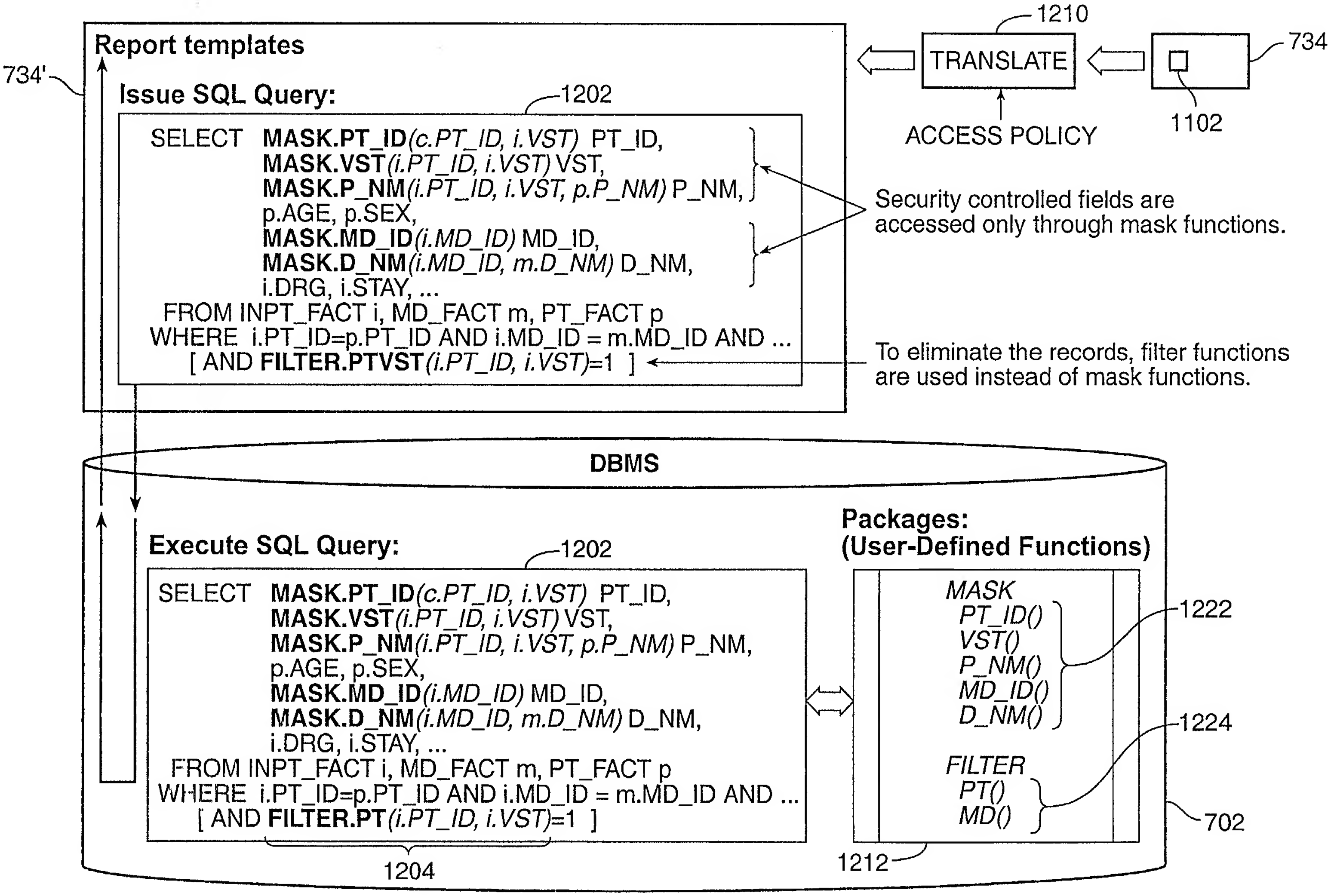
1102

FIG. 11

```
SELECT   MASK.PT_ID(c.PT_ID, i.VST)  PT_ID,
         MASK.VST(i.PT_ID, i.VST) VST,
         MASK.P_NM(i.PT_ID, i.VST, p.P_NM) P_NM,
        {MASK.AGE(i.PT_ID, i.VST, p.AGE) AGE, }
         p.SEX,
         MASK.MD_ID(i.MD_ID) MD_ID,
         MASK.D_NM(i.MD_ID, m.D_NM) D_NM,
         i.DRG, i.STAY, ...
  FROM INPT_FACT i, MD_FACT m, PT_FACT p
WHERE  i.PT_ID=p.PT_ID AND i.MD_ID = m.MD_ID AND ... ;
```

1302

FIG. 13



Report templates

Issue SQL Query:

```
SELECT  MASK.PT_ID(c.PT_ID, i.VST) PT_ID,
        MASK.VST(i.PT_ID, i.VST) VST,
        MASK.P_NM(i.PT_ID, i.VST, p.P_NM) P_NM,
        p.AGE, p.SEX,
        MASK.MD_ID(i.MD_ID) MD_ID,
        MASK.D_NM(i.MD_ID, m.D_NM) D_NM,
        i.DRG, i.STAY, ...
 FROM INPT_FACT i, MD_FACT m, PT_FACT p
WHERE  i.PT_ID=p.PT_ID AND i.MD_ID = m.MD_ID AND ...
    [ AND FILTER.PTVST(i.PT_ID, i.VST)=1  ]
```

Security controlled fields are accessed only through mask functions.

To eliminate the records, filter functions are used instead of mask functions.

DBMS

Execute SQL Query:

```
SELECT  MASK.PT_ID(c.PT_ID, i.VST) PT_ID,
        MASK.VST(i.PT_ID, i.VST) VST,
        MASK.P_NM(i.PT_ID, i.VST, p.P_NM) P_NM,
        p.AGE, p.SEX,
        MASK.MD_ID(i.MD_ID) MD_ID,
        MASK.D_NM(i.MD_ID, m.D_NM) D_NM,
        i.DRG, i.STAY, ...
 FROM INPT_FACT i, MD_FACT m, PT_FACT p
WHERE  i.PT_ID=p.PT_ID AND i.MD_ID = m.MD_ID AND ...
    [ AND FILTER.PT(i.PT_ID, i.VST)=1  ]
```

Packages:
(User-Defined Functions)

FIG. 12